EE524 EX6 Procedures

Image Rotation Kernel

1. Create KDF session
2. Define Rotate kernel in the *.cl file
3. Use a Sampler object defined at kernel file scope, in __constant memory
    a. Non-normalized coordinates
    b. Linear interpolation filter
    c. Clamp
4. Build Session and ensure kernel compiles successfully
5. Use Code Builder Analysis Input to assign Kernel Arguments
    a. create Images of type image2d_t for Input and Output
    b. use the **ee524_ex6_inputimg1.png** as input source
        i. Source Format: RGB-BGRA
        ii. Channel Data Type: CL_UNSIGNED_INT8
        iii. Channel Order: CL_RGBA
        iv. Depth = 1
        v. Array Size = 1
        vi. Row Pitch = 0
        vii. Slice Pitch = 0
    c. determine image width & height from File/Properties/Details
    d. experiment with various values of rotation angle ***theta*** (degrees)
6. Set Workgroup Size Definitions using X = #rows, Y = #columns
    a. Local size = 8,8,0
    b. Iterations = 1
7. Run the kernel
8. In CodeBuilder Run Results / Kernel Variables
    a. click image_out
        i. you should see your rotated image
    b. in lower right corner click the tiny “Compare Menu” icon
        i. a small window titled “Choose images to compare to:” should appear
        ii. select image_rotate::image_in
            1. this should show a two-pane display with input and output images
        iii. Note you can move cursor over image to view pixel R,G,B,A values in both images
9. Next try auto-tuning Local size using 20 iterations
    a. find fastest configuration and update Local size to use this value
10. Change Sampler to use CLK_ADDRESS_CLAMP_TO_EDGE
    a. Run and view output result image
11. Change Sampler to use CLK_FILTER_NEAREST
    a. Run and view output result image. Compare to previous output with LINEAR

Gaussian Blur 2D Convolution Kernel

1. Create KDF session
2. Define GaussBlur kernel in the *.cl file
   a. Note that this time the Sampler is a kernel function parameter
3. Build Session and ensure kernel compiles successfully
4. Use Code Builder Analysis Input to assign Kernel Arguments
   a. create Images of type image2d_t for Input and Output
      i. use the **ee524_ex6_inputimg1.png** as input source
         1. Source Format: RGB-BGRA
         2. Channel Data Type: CL_UNSIGNED_INT8
         3. Channel Order: CL_RGBA
         4. Depth = 1
         5. Array Size = 1
         6. Row Pitch = 0
         7. Slice Pitch = 0
      ii. determine image width & height from File/Properties/Details
   b. create a Sampler variable type and assign to the sampler Arg
      i. CL_ADDRESS_CLAMP_TO_EDGE
      ii. CL_FILTER_LINEAR
      iii. Do not check “Normalized Coordinates”
   c. Set Workgroup Size Definitions using X = #rows, Y = #columns
   d. Local size = 8,8,0
   e. Iterations = 1
5. Run the kernel
6. In CodeBuilder Run Results / Kernel Variables
   a. view the image_out result and compare to image_in
      i. use CTRL+ to zoom (or toolbar controls)
7. Use MATLAB **GaussBlurFilterCoeffs.m** to create a new set of filter coefficients
   a. Set linspace(-1.0, 1.0, 7)
   b. Ap = 1;
   c. sigma2 = 0.75
   d. copy the output comma-separated string from MATLAB console
8. paste into GaussBlur kernel file
   a. use as initialization values for a new __constant float gaussBlurFilter[ ] array variable.
   b. update the filterWidth variable
   c. comment out the 5x5 gaussBlurFilter /* … */
9. Build Session and ensure compiles correctly
10. Run kernel and compare output to input
   a. compare to previous 5x5 blurred result output